

UVG

Algoritmos y Programación Básica

Rodolfo Galdámez: 11040

Alejandra Reyes: 11231

Manual del usuario

Como primer paso hay que descargar el programa de Python, para este caso se utilizara la versión 2.7.

Puede ingresar al siguiente link:

<http://www.python.org/getit/>

Al ingresar vera las siguientes opciones:

Download Python

The current production versions are [Python 2.7.1](#) and [Python 3.2](#).

Start with one of these versions for learning Python or if you want the most stability, they're both considered stable production releases.

If you don't know which version to use, start with Python 2.7; more existing third party software is compatible with Python 2 than Python 3 right now.

For the MD5 checksums and OpenPGP signatures, look at the [detailed Python 2.7.1](#) page:

- [Python 2.7.1 Windows Installer](#) (Windows binary -- does not include source)
- [Python 2.7.1 Windows X86-64 Installer](#) (Windows AMD64 / Intel 64 / X86-64 binary [\[1\]](#) -- does not include source)
- [Python 2.7.1 Mac OS X 32-bit i386/PPC Installer](#) (for Mac OS X 10.3 through 10.6 [\[2\]](#))
- [Python 2.7.1 Mac OS X 64-bit/32-bit x86-64/i386 Installer](#) (for Mac OS X 10.6 [\[2\]](#))
- [Python 2.7.1 compressed source tarball](#) (for Linux, Unix or Mac OS X)
- [Python 2.7.1 bzipipped source tarball](#) (for Linux, Unix or Mac OS X, more compressed)

Asegúrese de descargar el programa de acuerdo a su computador.

También será necesario descargar Matplotlib y Numpy para poder lograr un interfaz grafico.

Puede ingresar al siguiente link:

<http://www.lfd.uci.edu/~gohlke/pythonlibs/>

Encontrara las siguientes opciones en cada uno de ellos:

Matplotlib is a 2D plotting library. A port to Python 3 is under way.

- [matplotlib-1.0.1.win-amd64-py2.5.exe](#) [8.5 MB] [Python 2.5] [64 bit] [Feb 04, 2011]
- [matplotlib-1.0.1.win-amd64-py2.6.exe](#) [8.5 MB] [Python 2.6] [64 bit] [Feb 04, 2011]
- [matplotlib-1.0.1.win-amd64-py2.7.exe](#) [8.5 MB] [Python 2.7] [64 bit] [Feb 04, 2011]
- [matplotlib-1.0.1.win32-py2.4.exe](#) [8.2 MB] [Python 2.4] [32 bit] [Feb 04, 2011]
- [matplotlib-1.0.1.win32-py2.5.exe](#) [8.2 MB] [Python 2.5] [32 bit] [Feb 04, 2011]
- [matplotlib-1.0.1.win32-py2.6.exe](#) [8.3 MB] [Python 2.6] [32 bit] [Feb 04, 2011]
- [matplotlib-1.0.1.win32-py2.7.exe](#) [8.3 MB] [Python 2.7] [32 bit] [Feb 04, 2011]
- [matplotlib-1.1.0.dev.win-amd64-py3.1.exe](#) [11.0 MB] [Python 3.1] [64 bit] [Mar 16, 2011]
- [matplotlib-1.1.0.dev.win-amd64-py3.2.exe](#) [11.0 MB] [Python 3.2] [64 bit] [Mar 16, 2011]
- [matplotlib-1.1.0.dev.win32-py3.1.exe](#) [10.8 MB] [Python 3.1] [32 bit] [Mar 16, 2011]
- [matplotlib-1.1.0.dev.win32-py3.2.exe](#) [10.8 MB] [Python 3.2] [32 bit] [Mar 16, 2011]

NuNumPy is a fundamental package needed for scientific computing with Python.

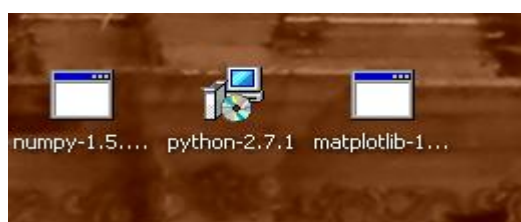
Note: these builds are not compatible with the official Scipy distributions.

Note: you may not redistribute the MKL builds unless you own an appropriate license from Intel.

- [numpy-1.5.1.win-amd64-py2.5-MKL.exe](#) [8.4 MB] [Python 2.5] [64 bit] [Feb 25, 2011]
- [numpy-1.5.1.win-amd64-py2.5.exe](#) [2.4 MB] [Python 2.5] [64 bit] [Nov 21, 2010]
- [numpy-1.5.1.win-amd64-py2.6-MKL.exe](#) [8.4 MB] [Python 2.6] [64 bit] [Feb 25, 2011]
- [numpy-1.5.1.win-amd64-py2.6.exe](#) [2.3 MB] [Python 2.6] [64 bit] [Nov 21, 2010]
- [numpy-1.5.1.win-amd64-py2.7-MKL.exe](#) [8.4 MB] [Python 2.7] [64 bit] [Feb 25, 2011]
- [numpy-1.5.1.win-amd64-py2.7.exe](#) [2.3 MB] [Python 2.7] [64 bit] [Nov 21, 2010]
- [numpy-1.5.1.win-amd64-py3.1-MKL.exe](#) [8.6 MB] [Python 3.1] [64 bit] [Feb 25, 2011]
- [numpy-1.5.1.win-amd64-py3.1.exe](#) [2.5 MB] [Python 3.1] [64 bit] [Nov 21, 2010]
- [numpy-1.5.1.win-amd64-py3.2-MKL.exe](#) [8.6 MB] [Python 3.2] [64 bit] [Feb 25, 2011]
- [numpy-1.5.1.win-amd64-py3.2.exe](#) [2.5 MB] [Python 3.2] [64 bit] [Dec 05, 2010]
- [numpy-1.5.1.win32-py2.5-MKL.exe](#) [6.6 MB] [Python 2.5] [32 bit] [Feb 25, 2011]
- [numpy-1.5.1.win32-py2.5.exe](#) [2.0 MB] [Python 2.5] [32 bit] [Nov 21, 2010]
- [numpy-1.5.1.win32-py2.6-MKL.exe](#) [6.8 MB] [Python 2.6] [32 bit] [Feb 25, 2011]
- [numpy-1.5.1.win32-py2.6.exe](#) [2.1 MB] [Python 2.6] [32 bit] [Nov 21, 2010]
- [numpy-1.5.1.win32-py2.7-MKL.exe](#) [6.8 MB] [Python 2.7] [32 bit] [Feb 25, 2011]
- [numpy-1.5.1.win32-py2.7.exe](#) [2.1 MB] [Python 2.7] [32 bit] [Nov 21, 2010]
- [numpy-1.5.1.win32-py3.1-MKL.exe](#) [6.9 MB] [Python 3.1] [32 bit] [Feb 25, 2011]
- [numpy-1.5.1.win32-py3.1.exe](#) [2.3 MB] [Python 3.1] [32 bit] [Nov 21, 2010]
- [numpy-1.5.1.win32-py3.2-MKL.exe](#) [6.9 MB] [Python 3.2] [32 bit] [Feb 25, 2011]
- [numpy-1.5.1.win32-py3.2.exe](#) [2.3 MB] [Python 3.2] [32 bit] [Dec 05, 2010]

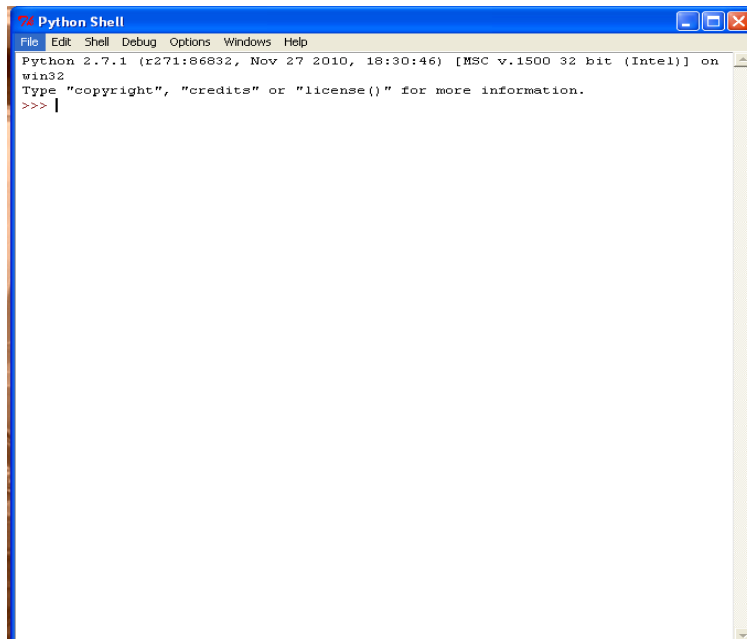
Asegúrese de que el link diga py2.7 ya que es el programa correspondiente a nuestra versión de Python. Usted verá varias opciones para Python 2.7, escoja la adecuada para la capacidad de su computadora.

Al terminar de bajar los 3 programas le aparecerán lo siguiente en su escritorio:



Primero debemos de instalar Python para no tener ningún problema con los demás programas. Estos se deben de instalar uno por uno.

Al finalizar la descarga de Python le aparecerá lo siguiente:



Al ingresar a IDLE se nos desplegara una ventanilla:

Nos encontramos en el Python Shell, lugar en donde se ejecutan los programas.

Para empezar a crear un programa siga los siguientes pasos:

- Clic en File
- New Window

Como la función lo indica, se le abrirá una nueva ventana, en esta es en donde usted debe realizar su programa.

Nota: Al momento de guardar su programa asegúrese de guardarlo con la extensión **.py**
Ejemplo: BlackJack.py

Juego de BlackJack

El objetivo del proyecto 2 es el de crear un juego, en este caso BlackJack, con el programa Python en donde pueden jugar únicamente 2 personas a la vez.

- Se creó una pequeña imagen que aparecerá al inicio del juego al ejecutar dicho programa.

```
print '*****BLACKJACK*****'

print ' *****      '
print '  *****      '
print '    *****      '
```

- Se crearon contadores para guardar cuantas veces a ganado un jugador.

```
#Desde Aqui empieza nuestro programa
# inicializamos todos los contadores
cont_jug1 = 0
cont_jug2 = 0
cont_empate = 0
main()    #llamamos a la funcion main() , sin nungun argumento
```

- Se importaron las funciones pylab y random.

```
#importamos pylab para la grafica de las estadisticas
from pylab import *
import random #importamos random para que nos de numeros al azar
```

- La función 'main' nos sirve para darle las primeras instrucciones de juego a los jugadores y para desplegarle las primeras cartas en orden aleatorio a cada uno de ellos.

```
#esta funcion nos sirve para dar las primeras cartas a los jugadores
def main():
    global nomj1 #hacemos Globales las Variables de los nombres
    global nomj2
    print "Este juego solo es de dos jugadores y la maquina va a repartir las ca
    print ' '
    print " Para Empezar a jugar introduce 1 y para salir 2  "
```

- Se definieron ambos jugadores con funciones. Utilizamos global para poderlas utilizar variables en otras funciones

```
#Esta funcion nos sirve para guardar los datos del jugador 1 y para preguntarle si quiere otro turno o para plantearse
def jug1(total,total2):
    global cont_jug2#Hacemos Globales los Contadores para Poderlas Utilizar en otras Funciones
    global cont_jug1

#Esta funcion nos sirve para guardar los datos del jugador 2 y para preguntarle si quiere otro turno o para plantearse
def jug2(total,total2):
    global cont_jug1
    global cont_jug2
```

- Se creó otra función para que en el momento de terminar el turno despliegue una pregunta para ver si desean otro turno.

```
#esta funcion nos sirve para terminar el turno en el que estan jugando
def terminar(total,total2):
    global cont_jug2
    global cont_jug1
    global cont_empate
    print 'Desean Otro turno (ingrese 1 ) o desean terminar el juego(ingrese 2) ?'
```

- Se creo una funcion en la cual al terminar una partida se le preguntara a los jugadores si quieren seguir jugando o salir. Al ingresar si se creara una nueva partida y al ingresar no desplegara unas graficas de estadisticas del juego.

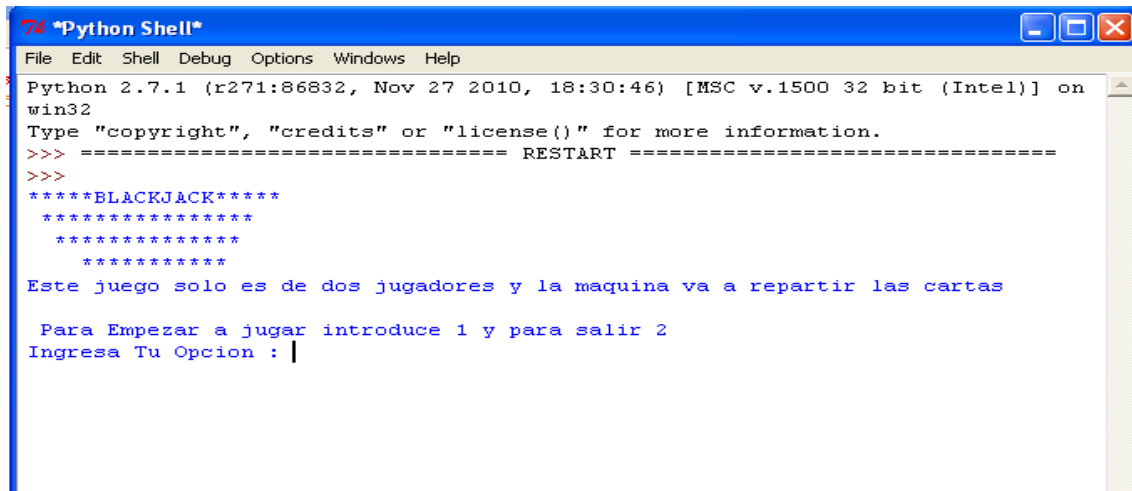
```
#Esta funcion nos sirve para preguntar si desean nu nuevo Juego
def Seguir_pre():
    z = raw_input( 'Quisiera a Empezar un Nuevo Juego si/no ?\n')

    if (z == 'si') or (z == 'SI') : #si desean jugar otro juego
        main()
    elif (z=='no') or (z== 'NO') :
        #esta parte es para graficar todos los juegos ganados y perdidos
        #grafica de barras
        print ' Gracias Por Jugar BLACKJACK'
        labels = ["Jugador 1 ", "Jugador2","Empates"]
        data = [cont_jug1 , cont_jug2 , cont_empate] #datos
        xlocations = array(range(len(data))+0.5 #arreglo para eje x
        width = 0.5
        xlim(0,10)
        ylim(0,3)
```

Dentro de estas definiciones utilizamos condiciones como 'if', 'and' y 'or'. Se utilizaron signos de desigualdad para que la computadora haga distintas funciones. Las repeticiones realizadas con la condición 'while', nos sirven para mantener un ciclo, el cual se interrumpirá en el momento de realizar lo contrario.

```
while res != '1' and res !='2': #Si la opcion es incorrecta se mete a un ciclo
    print ' Opcion Incorrecta, Ingrese un Dato Valido '
    print 'Desean Otro turno (Ingrese 1) o desean terminar el juego(Ingrese 2) ?'
    res=raw_input()
if (total>total2)and (total<21):
    print 'EL JUGADOR NO.1 GANOOOOOOOOOOO \n'
    cont_jug1+=1 #Le sumamos un punto al jug1
if (z == 'si') or (z == 'SI') : #si desean jugar otro juego , nos manda a la funcion main
    main()
```

Al ejecutar el programa (Clic en Run), el juego dará inicio, apareciendo así la siguiente ventanilla:

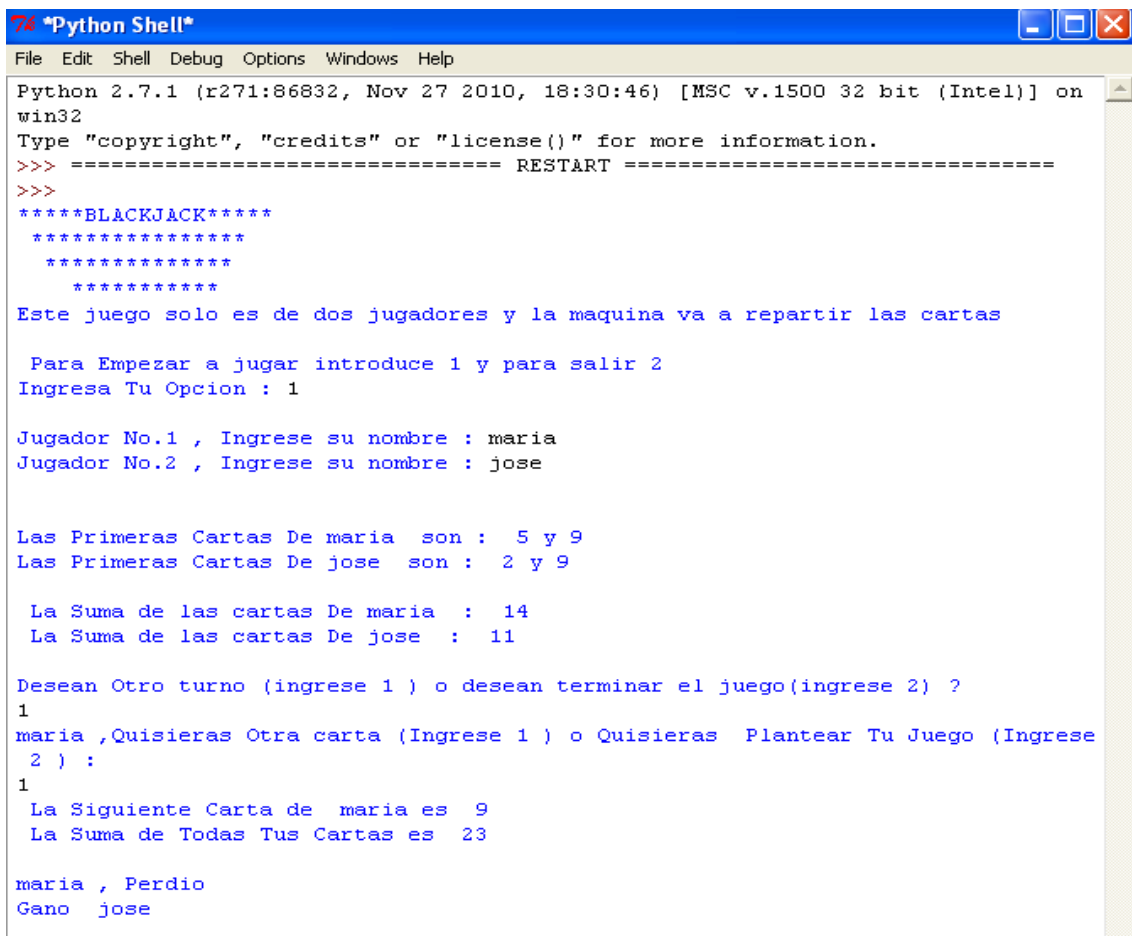


```
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
****BLACKJACK****
*****
*****
*****
Este juego solo es de dos jugadores y la maquina va a repartir las cartas

Para Empezar a jugar introduce 1 y para salir 2
Ingresa Tu Opcion : |
```

La partida se acabara automáticamente en el momento en el que uno de los jugadores obtenga 21 o más de 21.

Ejemplo:



```
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
****BLACKJACK****
*****
*****
*****
Este juego solo es de dos jugadores y la maquina va a repartir las cartas

Para Empezar a jugar introduce 1 y para salir 2
Ingresa Tu Opcion : 1

Jugador No.1 , Ingrese su nombre : maria
Jugador No.2 , Ingrese su nombre : jose

Las Primeras Cartas De maria son : 5 y 9
Las Primeras Cartas De jose son : 2 y 9

La Suma de las cartas De maria : 14
La Suma de las cartas De jose : 11

Desean Otro turno (ingrese 1 ) o desean terminar el juego(ingrese 2) ?
1
maria ,Quisieras Otra carta (Ingrese 1 ) o Quisieras Plantear Tu Juego (Ingrese 2 ) :
1
La Siguiente Carta de maria es 9
La Suma de Todas Tus Cartas es 23

maria , Perdio
Gano jose
```

Al terminar la partida le aparecerá la opción de si desea jugar si/no, escribir si, se jugara una nueva partida, pero al escribir no, desplegara los gráficos del rendimiento de ambos jugadores en las partidas realizadas.

Ejemplo: Si

```
maria , Perdio
Gano jose

Quisiera a Empezar un Nuevo Juego si/no ?
si
Este juego solo es de dos jugadores y la maquina va a repartir las cartas

Para Empezar a jugar introduce 1 y para salir 2
Ingresa Tu Opcion : 1

Jugador No.1 , Ingrese su nombre : maria
Jugador No.2 , Ingrese su nombre : jose

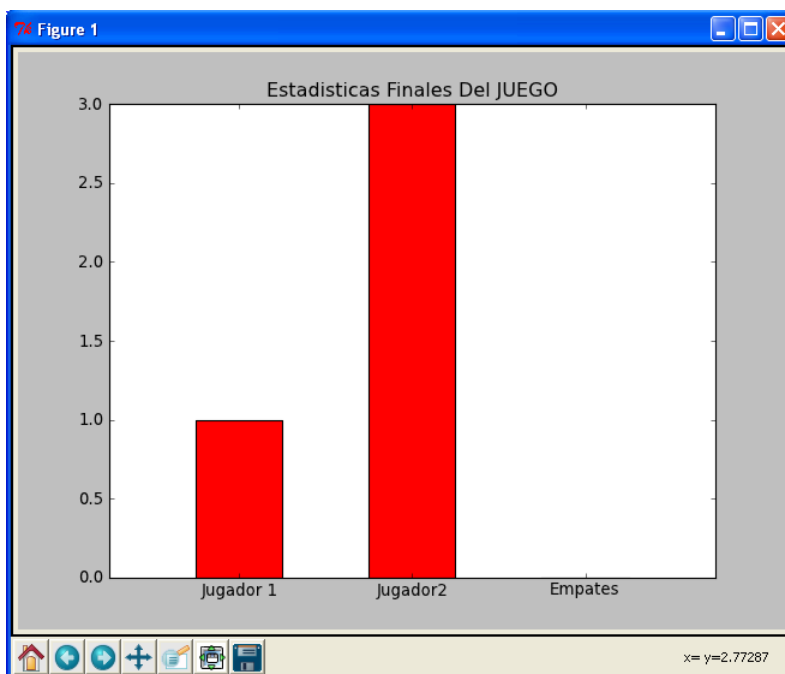
Las Primeras Cartas De maria son : 7 y 9
Las Primeras Cartas De jose son : 5 y 1

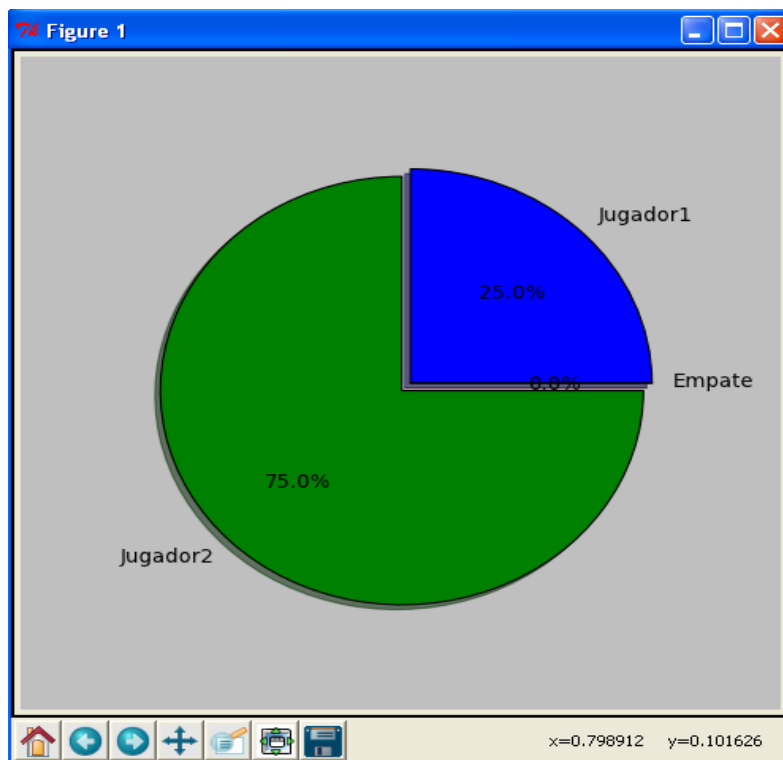
La Suma de las cartas De maria : 16
La Suma de las cartas De jose : 6
```

Ejemplo: no

```
maria , Perdio
Gano jose

Quisiera a Empezar un Nuevo Juego si/no ?
no
Gracias Por Jugar BLACKJACK
```





Errores o Comentarios

Si tuviste algún tipo de error en la ejecución del programa puedes mandarnos un correo a:

- chofogf_17@hotmail.com
- gal11040@uvg.edu.gt
- Rey11231@uvg.edu.gt

Si deseas dejar un comentario, también puedes mandar tu opinión a :

- gal11040@uvg.edu.gt
- rey11231@uvg.edu.gt